第4章 ADuC7026開発ツールの使いかた

── 二つの統合開発環境とPLA設計ツール

松添信宏

ここでは，付属基板の活用に不可欠なソフトウェア開発ツールのうち，統合開発環境の「KEIL Development Suite for ARM」と「IAR Embedded Workbench Kickstart for ARM」の使いかたを解説する．付属CD-ROMからツールをインストールし，ADuC7026向けのソフトウェア開発を行い，実際に動作させる．手順を具体的に示すので，同じように操作すればソフトウェア開発を体験できる．（編集部）

本誌付属のCD-ROMには，米国Analog Devices社のQuickStart Plus開発システムに含まれるソフトウェア一式が収録されています．本稿では統合開発環境（IDE）のセットアップとデバッグ方法，およびPLA設計ツール（PLA Tool）の使いかたについて解説します．

QuickStart Plus開発システムは，付属CD-ROMのsoft\adi\ADuC702xV1.0フォルダに収録されています[注1]．ディレクトリ構成を**図1**に示します．また，本稿の説明で使用するプロジェクト・ファイルとソース・コードは，articles\0603_a04\dataフォルダに収められています．

QuickStart Plus開発システムには，**表1**のようなツールが収録されています．

1）統合開発環境（IDE）

Keil Software社（2005年11月に買収され，現在は英国ARM社）の「KEIL Development Suite for ARM」[注2]とスウェーデンIAR Systems社の「IAR Embedded Workbench Kickstart for ARM」[注3]という2種類の統合開発環境の無償評価版が収録されています．また，フリーのツール・チェーンであるGNUのGCC ARMも収録されています．

二つの統合開発環境はいずれもターゲット・デバイスとしてADuC7000シリーズをサポートします．これらの評価版は実質無期限で使用できますが，コード・サイズに制限があります．

2）ダウンローダ

Analog Devices社のダウンローダARMWSD[注4]は，ADuC7026のUARTポート経由でIntel-HEX形式のコード・イメージをダウンロードするためのツールです．ダウンロード・ケーブルには市販のEIA-232（RS-232-C）用Dサブ9ピン・ケーブルが使えます．

3）PLA設計ツール

PLA Tool[注5]はプログラマブル・ロジック・アレイ（PLA）用のGUIソフトウェアです．このソフトウェアは，ADuC7000シリーズが内蔵するPLAのコンフィグレーシ

```
ADuC702xV1.0        QuickStart開発システム[setup.exe]
├ADuC702x
│ ├code             サンプル・コード集
│ ├documentation    ドキュメント集
│ ├download         シリアル・ダウンローダ[ARMWSD.exe]
│ ├help             ライブラリ・ヘルプ
│ ├platool          PLAツール[PLATool.exe]
│ └USBDriver
├IAR                IAR社統合開発環境[ewarm-ks-430a-2.exe]
├Keil               Keil社統合開発環境[KARM242a.exe]
└System32
```

図1　QuickStart Plus 開発システムの内容

注1：本稿執筆時点におけるQuickStart開発システムのバージョンは1.0であるが，まもなくアップデート版が公開される予定である．以下のサイトからダウンロードできる．
ftp://ftp.analog.com/pub/MicroConverter/ADuc702xV1.1/
注2：動作環境はWindows 98/NT/2000/XP，128Mバイト以上のメイン・メモリ，30Mバイト以上のハード・ディスク容量．
注3：動作環境はWindows 95/98/Me/2000/XP，128Mバイト以上のメイン・メモリ，100Mバイト以上のハード・ディスク容量．
注4：動作環境はWindows 2000/XP.
注5：動作環境はWindows 2000/XP.

表1　QuickStart Plus開発システムに収録されるツール

ソフトウェア名	ファイル名	機　能	ソース・レベル・デバッグ	制　限
KEIL Development Suite for ARM Version 2.42 Evaluation Software	KARM242a.exe	エディタ，アセンブラ，コンパイラ，リンカ	●シミュレータ ●JTAGポート経由（USB-JTAGエミュレータを使用）	コンパイラ，リンカ，デバッガが扱えるコードは16Kバイト以下
IAR Embedded Workbench Kickstart for ARM v4.30A	ewarm-ks-430a-2.exe	エディタ，アセンブラ，コンパイラ，リンカ	●シミュレータ ●JTAGポート経由（USB-JTAGエミュレータを使用） ●UART経由ROMモニタ・デバッグ	コンパイラ，リンカ，デバッガが扱えるコードは32Kバイト以下
ARMWSD	ARMWSD.exe	ダウンローダ		
PLA Tool	PLATool.exe	PLA設計ツール		

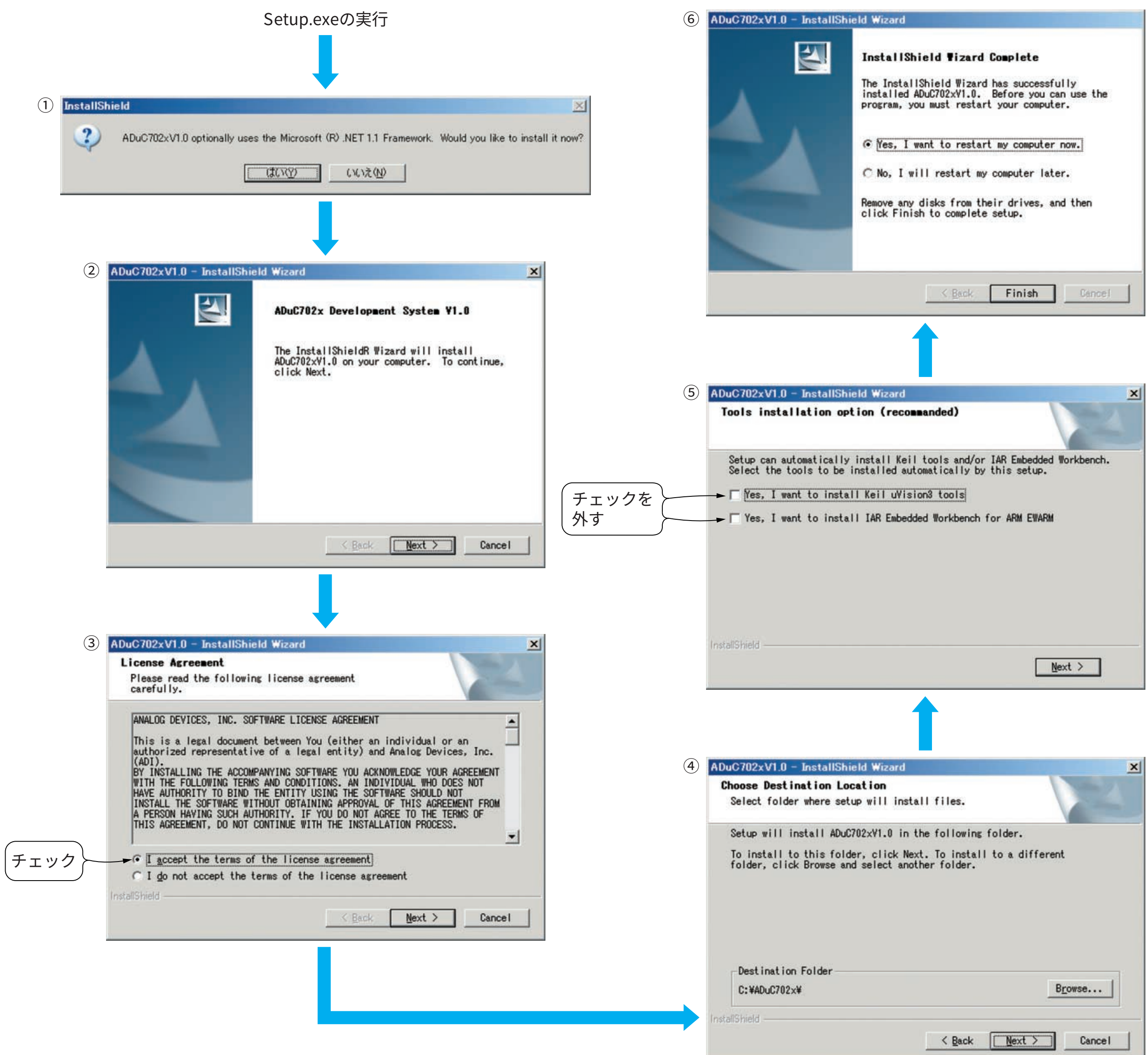


図2　ユーティリティのインストール手順

ョンをマウス操作だけで行うことができます．

このソフトウェアはアセンブリ言語またはC言語のソース・コードを生成します．生成されたソース・コードをIDEのプロジェクトにコピーして使います．

1．ユーティリティのインストール

初めにユーティリティ・ソフトウェアとサンプル・コード，ADuC7000シリーズのドキュメントなどをインストールします．図2にセットアップの手順を示します．

付属CD-ROMのsoft\adi\ADuC702xV1.0フォルダにあるsetup.exeをダブルクリックし，セットアップ・プログラムを実行します．

米国Microsoft社の.NET Frameworkがインストールされていない場合は，図2の①のメッセージが表示されます．これはPLA Toolランタイム・エンジンとして使われるので，［はい(Y)］ボタンをクリックしてインストールします．すでに.NET Frameworkがインストールされている場合は何も表示されません．

ADuC702xV1.0のインストーラが起動したら，まずは［Next >］ボタンをクリックします（図2の②）．すると，ライセンス契約（License Agreement）の画面が表示されます．内容を確認し，「I accept the terms of the license agreement」を選んだうえで，［Next >］ボタンをクリックします（図2の③）．

Choose Destination Location画面では，インストール先ディレクトリを指定します．ここではデフォルト（C:\ADuC702x\）のまま［Next >］ボタンをクリックします（図2の④）．

Tools installation option画面では，インストールするツールの選択を行います．本稿ではツールのインストールは後で行いますので，チェックを外してから［Next >］ボタンをクリックします（図2の⑤）．

ユーティリティ類のインストールはここで終わりです．［Finish］ボタンをクリックし，コンピュータを再起動します．

インストールが完了すると，C:\ADuC702xフォルダにサンプル・コード集，ドキュメント，ARMWSD，PLA Toolがインストールされます．以後，ARMWSDとPLA Toolは，Windowsの「スタート」メニューから「すべてのプログラム」→「ADuC702x」で起動できるようになります．

2. KEIL Development Suiteの使いかた

ここでは統合開発環境の「KEIL Development Suite

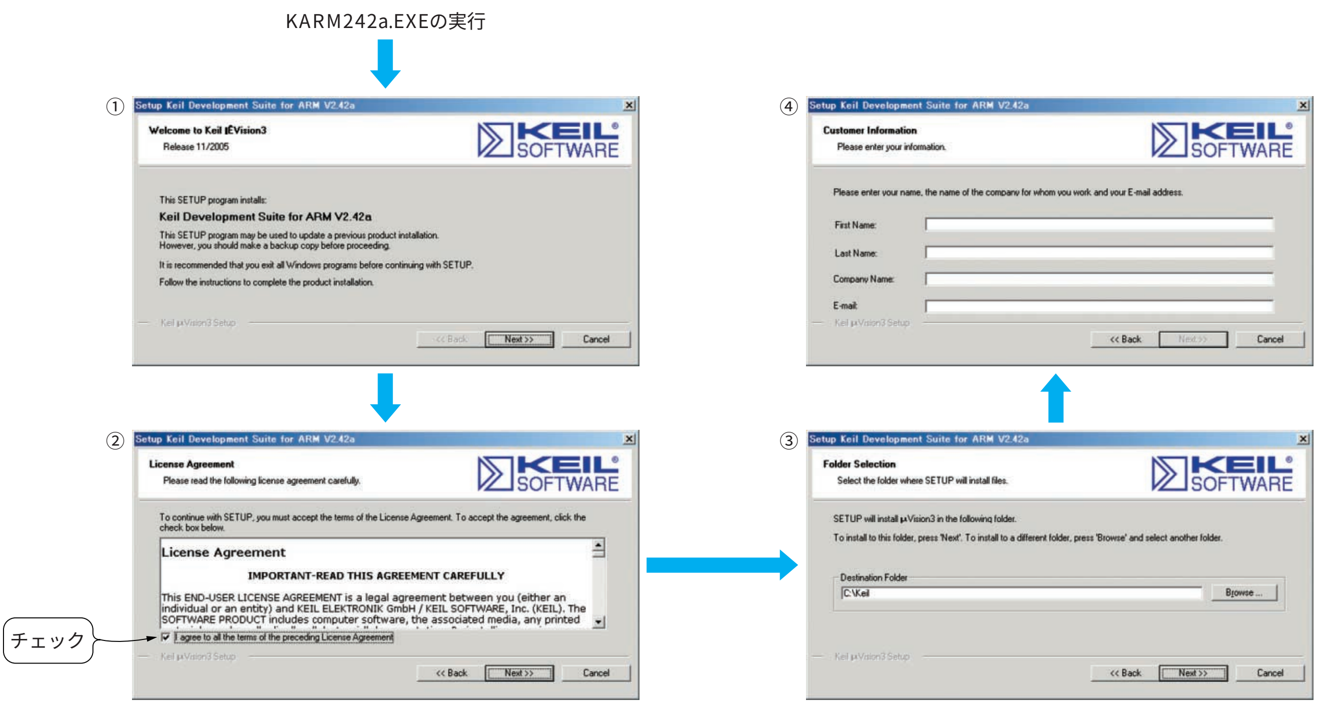


図3　KEIL Development Suite for ARMのインストール手順

for ARM」の使いかたを解説します．ツールのインストールを行った後，ソフトウェアをC言語で記述し，コンパイルおよびビルドします．コンパイルしたプログラムはARMWSDを使って付属基板にダウンロードし，実動作を確認します．

● インストール

KEIL Development Suite for ARMのインストール手順を図3に示します．

まず，付属CD-ROMのsoft \ adi \ ADuC702xV1.0 \ Keilフォルダにある KARM242a.EXEを起動します．セットアップ画面が表示されるので［Next >>］ボタンをクリックし

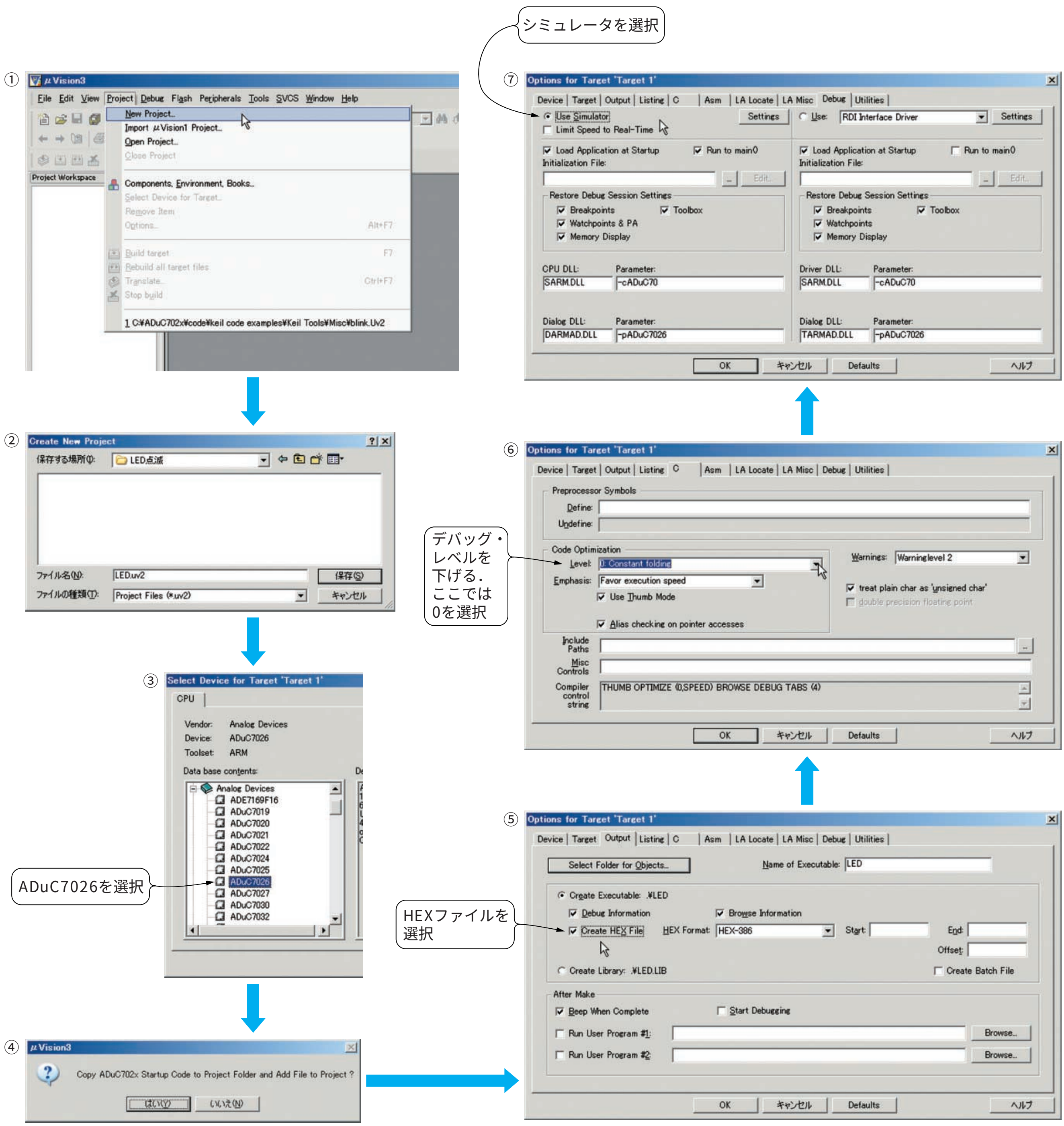


図4　KEIL Development Suite for ARMのプロジェクトの作成手順

ます(**図3**の①).

License Agreement画面が表示されます．内容を確認のうえ，「I agree to all the terms of the preceding License Agreement」にチェックを入れ，［Next >>］ボタンをクリックします(**図3**の②).

Folder Selection画面では，インストール先ディレクトリを指定します．ここではデフォルトのC:\Keilのまま［Next >>」ボタンをクリックします(**図3**の③).

最後に，ユーザ情報を入力して［Next >>］ボタンをクリックすると，インストールは完了です(**図3**の④).

● ソース・コードの準備

最初に，作業用のフォルダを用意します．Windowsのエクスプローラで作業フォルダを作成してください．ここでは，C:\ADuC702x\codeを作業フォルダとします．

作業フォルダに，ソフトウェアのソース・コードを用意します．統合開発環境を使ってソース・コードを作成できます．今回は手間を省くため，あらかじめ用意されているソース・コードを使用します．付属CD-ROMのarticles\0603_a04\dataフォルダに収められているDWM_ADuC7026_コードというフォルダを，フォルダごと作業フォルダ(C:\ADuC702x\code)にコピーします．

● プロジェクトの作成

Windowsの「スタート」メニューから「すべてのプログラム」→「Keil uVision3」を選択します．

まず，設計データを管理するためのプロジェクトを作成します．統合開発環境では，ターゲット・デバイスの選定，ソース・コード，オブジェクト・ファイルなどをプロジェクトという単位で管理します．手順を**図4**に示します．新規プロジェクトの作成では，「Project」→「New Project」を選択します(**図4**の①).

プロジェクトの保存先を指定します．先ほど用意したC:\ADuC702x\Code\DWM_ADuC7026_コード\LED点滅というフォルダをプロジェクトの保存先にします．プロジェクト名を「LED.uv2」として，［保存(S)］ボタンをクリックします(**図4**の②).

デバイス選択の画面が表示されます．「Analog Devices」の下にある「ADuC7026」を選択し，［OK］ボタンをクリックします(**図4**の③)．µVision3は，このデバイス・データをもとにスタートアップ・ファイルを自動生成します．

スタートアップ・ファイルをプロジェクトに追加するかどうかを聞かれるので，［はい(Y)］ボタンをクリックします(**図4**の④).

「Project」→「Options for Target“ Target1 ”」を選択す

4

Column JTAGエミュレータ「mIDAS-Link」について

ROMモニタを使った実機におけるソース・レベル・デバッグの問題として，ROMモニタがターゲットのリソースを使う点があります．モニタ・プログラムがフラッシュ・メモリの一部とUARTポートを占有してしまうため，デバッグ中はこれらをユーザ・アプリケーションで使うことができません．

ARM7TDMIコアは，デバッグ・インターフェース・モジュールを搭載しています．JTAGを使うことにより，デバイスをボードに実装した状態，かつ実速度でプログラムを実行し，デバッグを行えます．デバッグにはJTAGインターフェース専用のピンを使うため，ユーザ・プログラムはADuC7026のすべてのリソースを利用できます．

QuickStart Plus開発システムには，JTAGエミュレータとして「mIDAS-Link」が付属します(**写真A**)．これはドイツSegger Microcontroller Systeme社(http://www.segger.com/)のJLINKと同じものです．デバッグ・ホストのパソコンにはUSBで接続します．「mIDAS-Link」は，本稿で紹介した「µVision3」や「IAR Embedded Workbench」に加えて，ARM社の「RealView」，「ADS」などのRDI準拠のデバッグ・ホストにも対応しています．

なお，Analog Devices社から「mIDAS-LINK」を付属したQuick Start Plus開発システムがDesign Wave Magazine読者キャンペーン価格で提供されます．詳細は本誌の広告ページをご参照ください．



写真A　QuickStart Plus 開発システムに付属のJTAGエミュレータ
左がJTAGエミュレータの「mIDAS-Link」．右はQuickStart Plus開発システムに付属のボード．ほかに，電源やシリアル・ダウンロード・ケーブルなどが付属する．

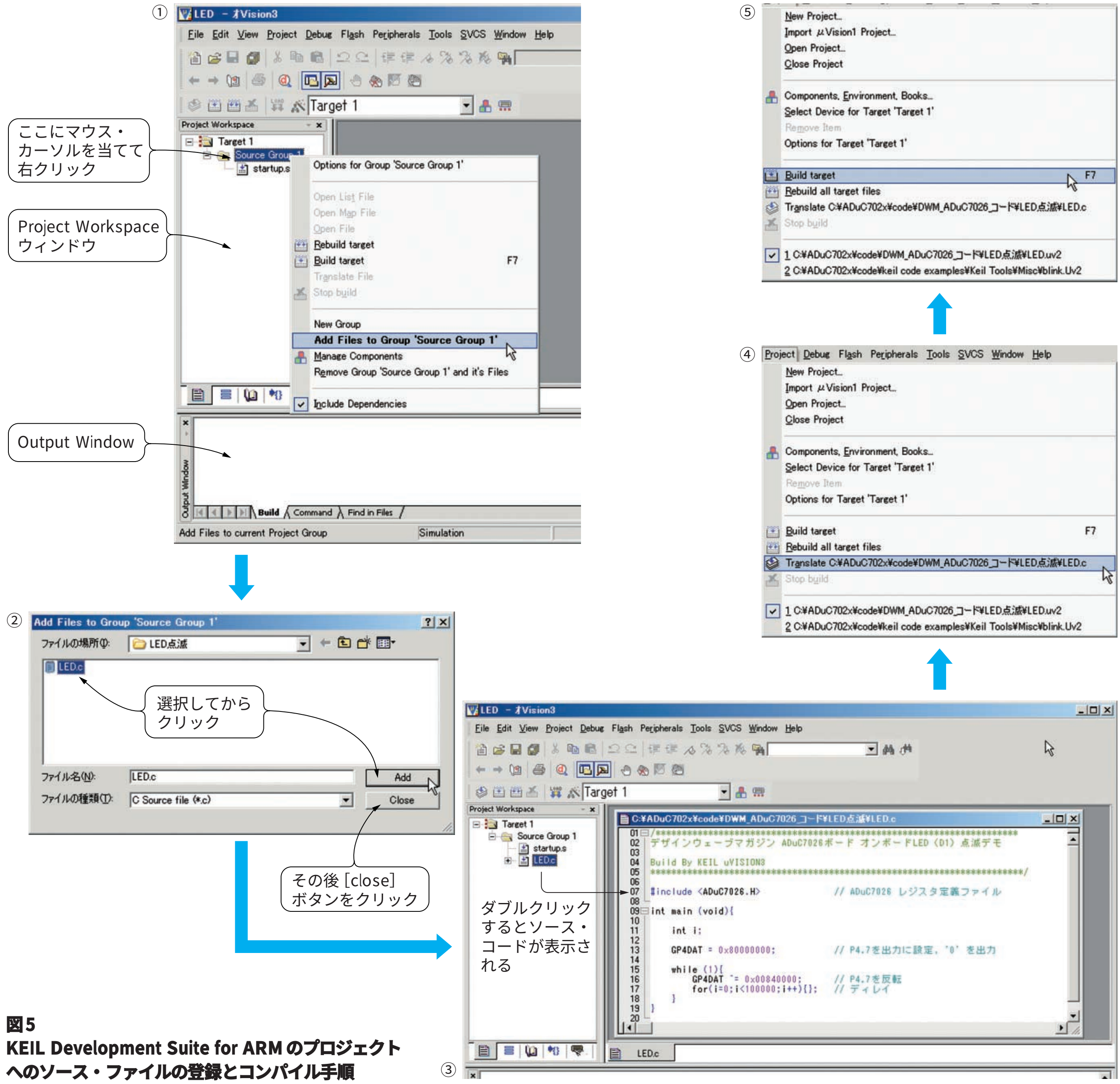


図5
KEIL Development Suite for ARMのプロジェクトへのソース・ファイルの登録とコンパイル手順

ると，オプション設定画面が表示されます．Outputタブをクリックし，「Create HEX File」にチェックを入れます．HEXファイルはダウンローダで使用するファイル形式です（図4の⑤）．また，ソース・レベル・デバッグを行う場合は，Cタブにある「Code Optimization」の「Level」を下げておくとトレースしやすくなります（図4の⑥）．次に，Debugタブを選択し，デバッグの方法を指定します（図4の⑦）．ここでは，シュミレータを使う「Use Simulator」を選択してください（なお，デバッグを実機で行う場合は「Use RDI Interface Driver」を選択する．この場合，JTAGエミュレータが別途必要）．これでオプションの設定は終わりです．［OK］ボタンをクリックしてオプション画面を終了します．

● ソース・ファイルの登録とコンパイル

作成したプロジェクトにソース・コードを登録し，コンパイルを行います．手順を図5に示します．

ソース・ファイルをプロジェクトに追加するためには，Project Workspaceウィンドウの「Target 1」の下にある

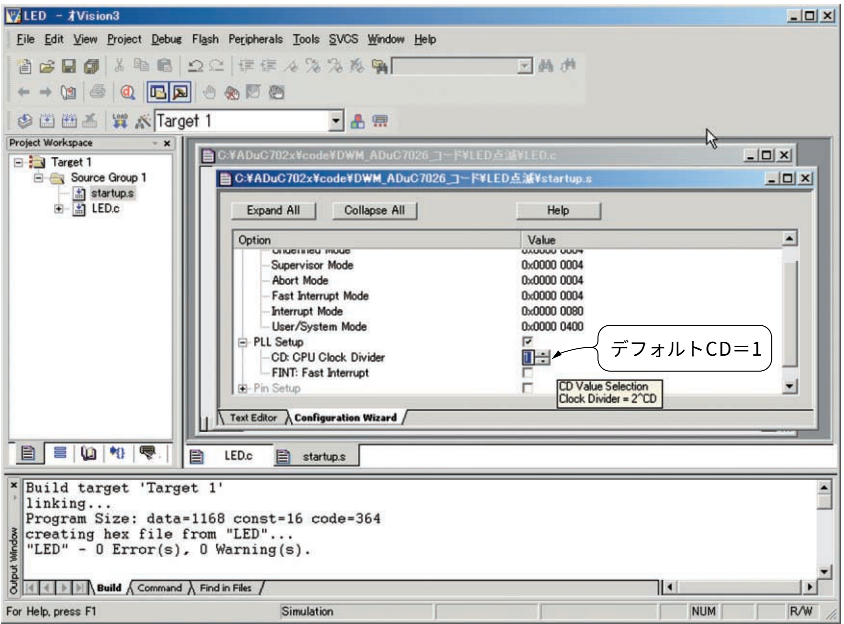


図6　スタートアップ・ファイル

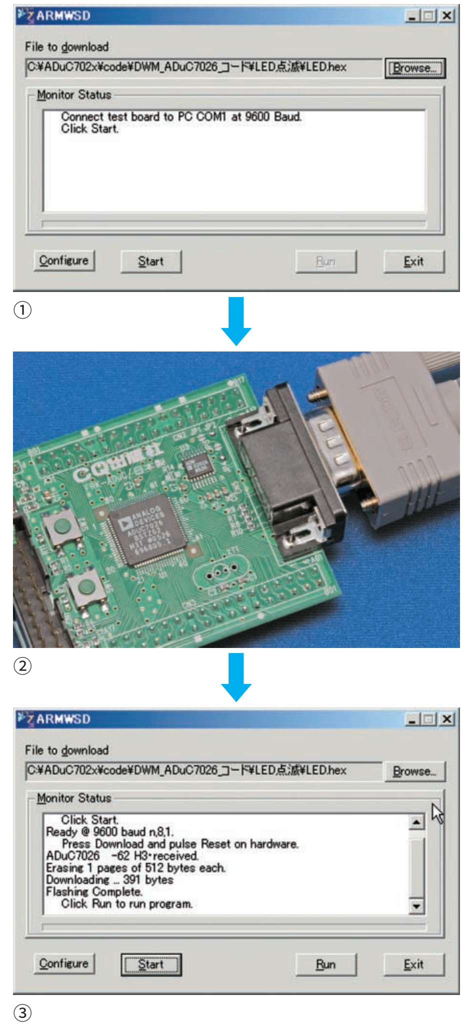


図7　ソフトウェアのデバイスへの書き込み手順

「Source Group 1」の文字の上にマウス・カーソルを当て，ここで右クリックして表示されるメニューから「Add Files to Group 'Source Group 1'」を選択します（図5の①）．

ファイル選択画面が表示されます．作業フォルダにあるLED.cファイルを選択し，［Add］ボタンをクリックし，その後［Close］ボタンをクリックします（図5の②）．

プロジェクトに登録されているソース・ファイルは，Project Workspaceウィンドウにツリー表示されます．LED.cが表示されているかどうかを確認してください．Project WorkspaceウィンドウのLED.cをダブルクリックすると，LED.cのソース・コードが表示されます（図5の③）．このコードは，付属基板のLED（D1）を点滅させるものです．

LED.cをコンパイルします．Project WorkspaceウィンドウのLED.cをクリックして選択した後，「Project」→「Translate ...」を選択します（図5の④）．これでコンパイルが始まります．エラーがなければ，Output Windowに「0 Error（s），0 Warning（s）」と表示されます．

次に，ダウンロードするROMイメージを作成します．「Project」→「Build Target」を選択して，ビルドを実行します（図5の⑤）．ビルドするとOutput Windowにプログラム・サイズが表示され，作業フォルダにIntel-Hex形式のROMイメージ（LED.hex）が作成されます．

ビルドの際に，必要に応じてスタートアップ・ファイルの設定を変更します．スタートアップ・ファイル（startup.s）は，スタック・サイズやCPUクロックを設定します．これは，Project Workspaceの「startup.s」をダブルクリックすると表示されます．スタートアップ・ファイルはテキストで直接変更することができますが，Configuration Wizardタブを選ぶとGUIが利用できます．「CPU Clock Divider」の設定がデフォルトのCD＝1に設定（22MHz動作）されていることが確認できます（図6）．

● ソフトウェアのデバイスへの書き込み

生成されたROMイメージのダウンロードを行います．手順を図7に示します．

Windowsの「スタート」から「すべてのプログラム」→「ADuC702x」→「ARMWSD」を選択し，ARMWSDを起

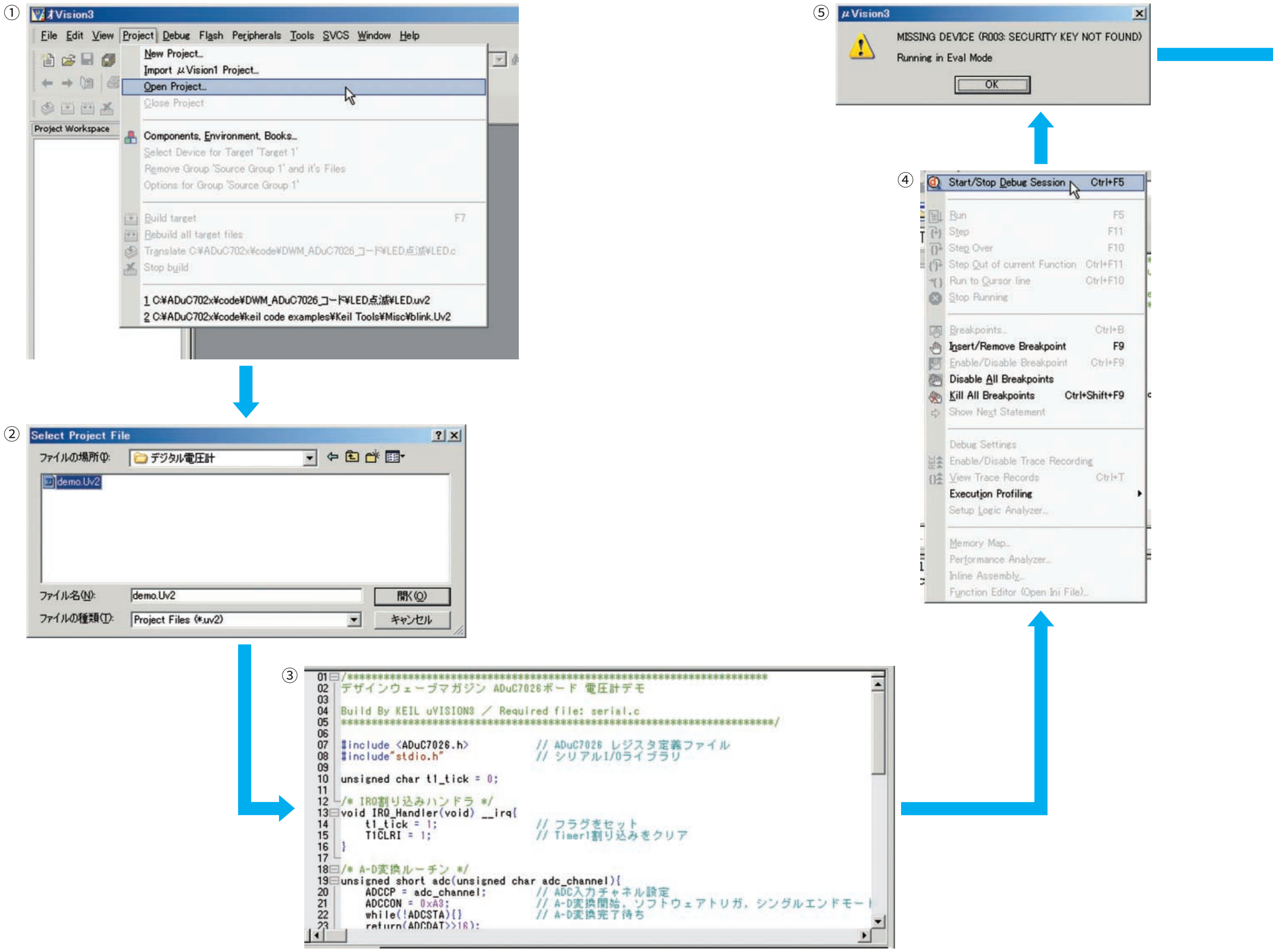


図8 KEIL Development Suite for ARMによるソース・レベル・デバッグの手順

動します．ARMWSDの[Browse]ボタンをクリックし，ダウンロードするHEXファイルを選びます．ビルドで作成されるHEXファイルは，作業フォルダ(ここでは，C:\ADuC702x\Code\DWM_ADuC7026_コード\LED点滅)に保存されます(図7の①)．

本誌付属基板に電源を入れ，パソコンとシリアル・ケーブルで接続します(図7の②)

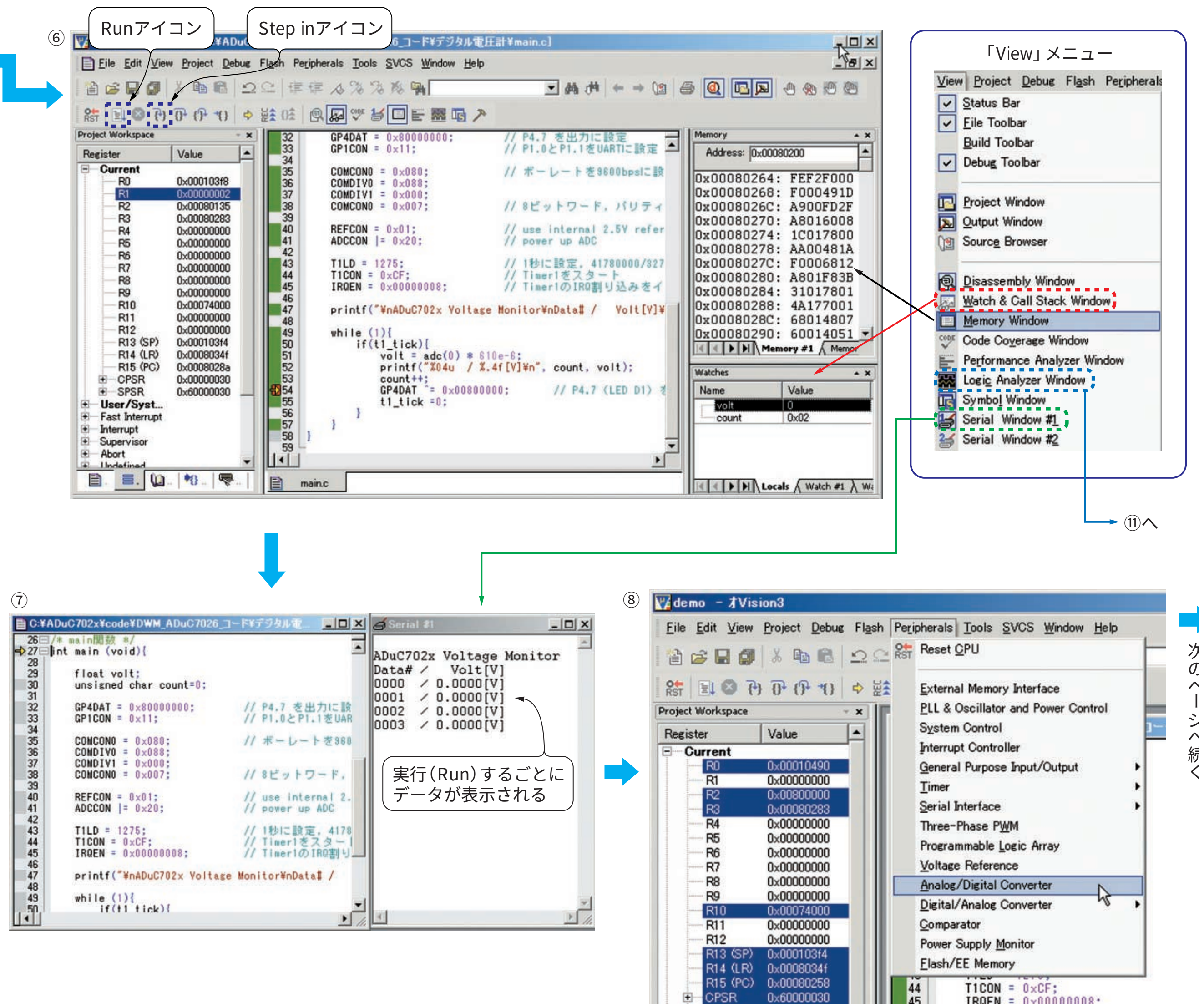
ADuC7026をダウンロード・モードに設定します．設定方法はBM(ブートモード)を“L”にした状態でリセットをかけます．本誌付属基板ではスイッチSW2を押したままの状態で，スイッチSW1を押して離し，その後スイッチSW2を離すという手順になります．

ADuC7026をダウンロード・モードに設定した後，ARMWSDの[Start]ボタンをクリックすると，ダウンロードが開始されます(図7の③)．ダウンロード終了後，リセット(SW1)をかけるとプログラムが実行され，オンボードのLEDが点滅動作します．

最後に，µVision3の画面で「Project」→「Close Project」を選択し，プロジェクトを終了します．

● ソース・レベル・デバッグ

µVision3では，JTAG経由またはシミュレータによるソース・レベル・デバッグを行うことができます．JTAG経由のデバッグにはJTAGエミュレータ(mIDAS-Linkなど)



次のページへ続く

が必要です．ここではシミュレータを使ってデバッグを実行します．説明のため，ここではあらかじめ用意されたプロジェクトを使い，デバッガのシリアル通信モニタとロジック・アナライザで動作を確認します．図8に手順を示します．

1）プロジェクトの用意

まず，あらかじめ用意してあるプロジェクトを開きます．プロジェクトを開くには「Project」→「Open Project」を選択します（図8の①）．

プロジェクト・ファイル（拡張子.uv2）を指定する画面が表示されるので，C:\ADuC702x\code\DWM_ADuC7026_コード\デジタル電圧計フォルダにあるdemo.Uv2を選択し，［開く］ボタンをクリックします（図8の②）．

Project Workspaceウィンドウにソース・コードの一覧が表示されていることを確認します．「Source Group 1」の下にあるmain.cをダブルクリックすると，ターゲットのコードが表示されます（図8の③）．「Project」→「Build Target」でプロジェクトをビルドします．

2）ブレークポイントの設定とステップ実行

次に，「Debug」→「Start/Stop Debug Session」でデバッグ・セッションを開始します（図8の④）．ここで評価バージョンであるメッセージが表示されます．エラーではありませんので，［OK］ボタンをクリックします（図8の⑤）．

ソース・コードの行番号の左側をダブルクリックすることでブレークポイントを設定できます（図8の⑥，「Debug」→「Insert/Remove Breakpoint」を選択してもよい）．プログラムは，「Debug」メニューから「Run」や「Step」などを選択することで実行できます．ツール・バーのアイコン（

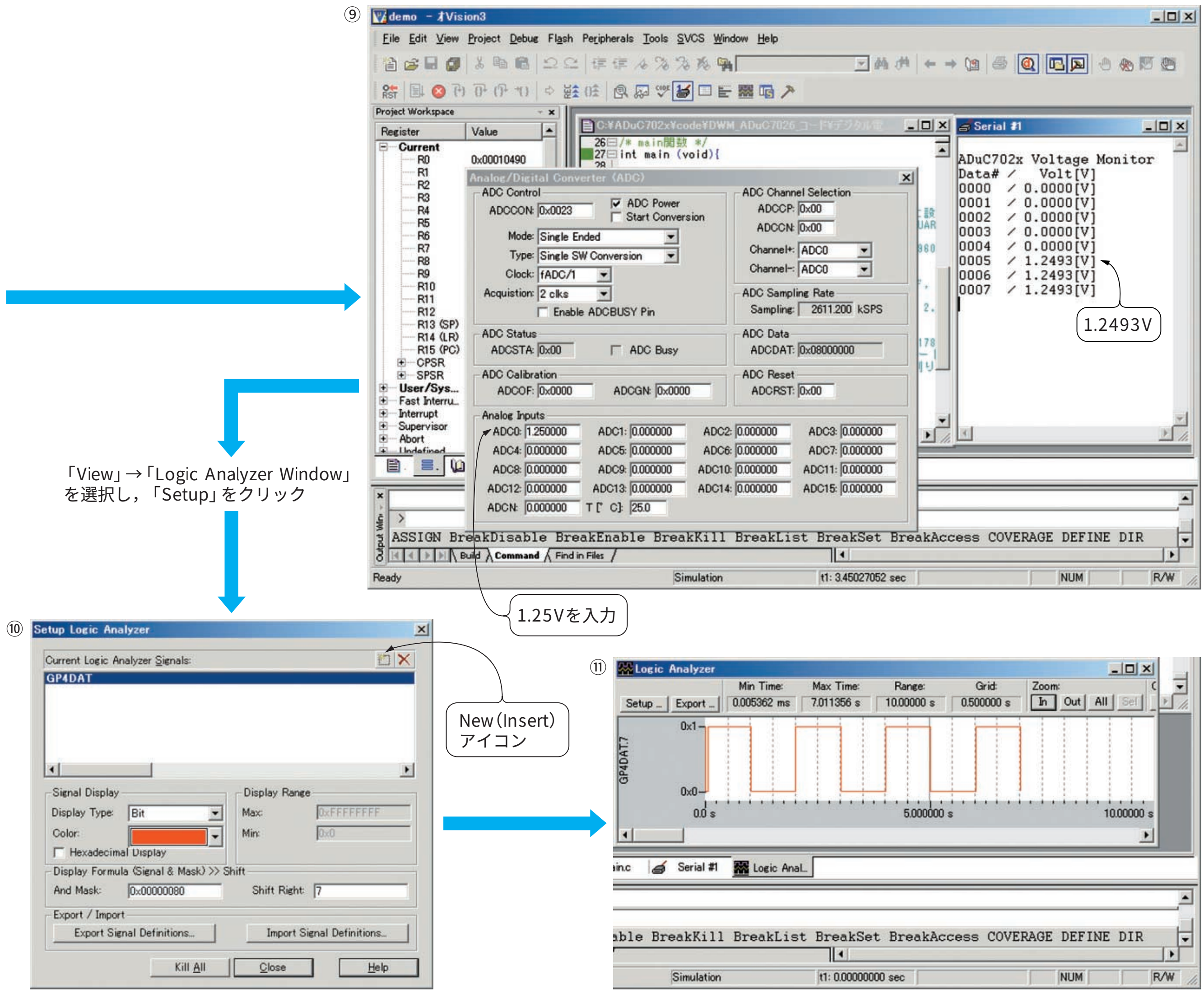


図8　KEIL Development Suite for ARMによるソース・レベル・デバッグの手順（つづき）

や など）をクリックしても実行します．

また，「View」メニューから，Memoryウィンドウ（搭載するフラッシュ・メモリとRAMの値を見ることができる．直接，値を書き換えられる）やWatchesウィンドウ（変数やネストされた関数呼び出しの状態を見ることができる．直接，値を書き換えられる）を呼び出せます．

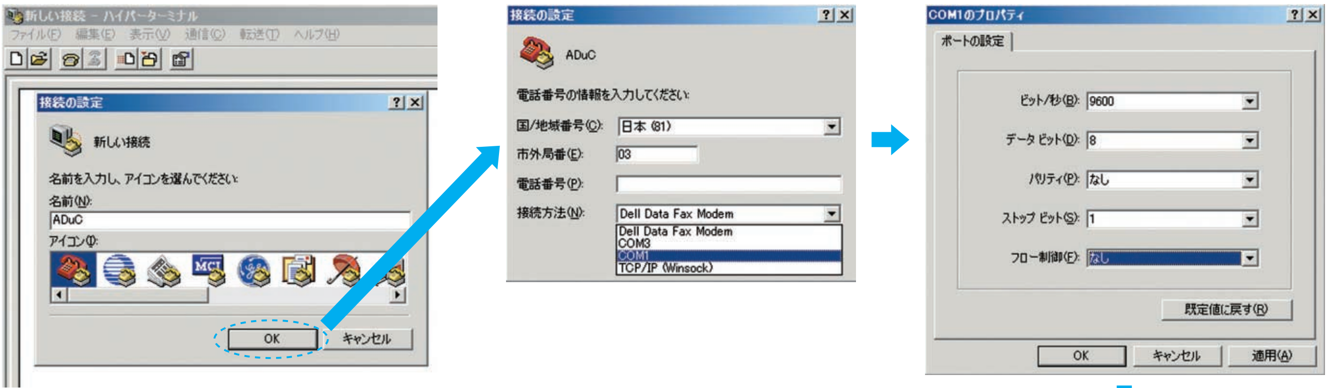
µVision3のデバッガには，シリアル通信のターミナル機能があります．「View」→「Serial Window #1」を選択すると，UARTの通信データを表示できます（図8の⑦）．まだA-Dコンバータの入力データを準備していないため，変換結果は0Vです．

3）デバイス専用GUIによるデバッグ

ADuC7026が内蔵する周辺I/Oは，メモリにマッピングされているので，メモリ・ウィンドウからその状態を見ることができます．また，µVision3にはデバイス専用のGUI画面も用意されています．デバイス専用のGUI画面は，「Peripherals」から呼び出します．

「Peripheral」→「Analog/Digital Converter」を選択して，A-Dコンバータの画面を表示させます（図8の⑧）．シミュレータを用いると，A-Dコンバータへの入力信号を設定し，計算結果を確認することができます．「Analog Inputs」のADC0のボックスに1.25を入力してみます（図8の⑨）．

これでA-D変換結果を電圧値としてモニタすることがで



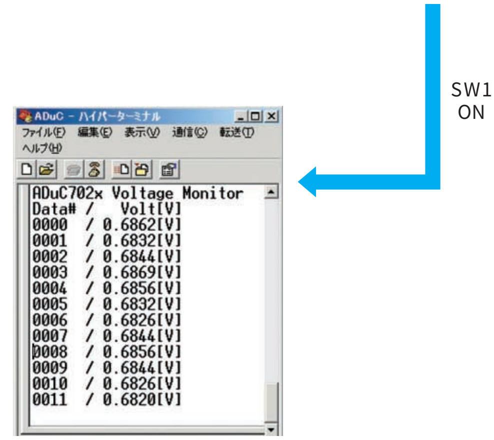


図9　ハイパーターミナルの設定

きるようになります．コンバータの変換結果はHEX値でADCDATレジスタに書き込まれます．プログラムを実行(Run)すると，シリアル通信のモニタ画面に1.2493［V］と表示されるのを確認できます(図8の⑨)．

次に，ロジック・アナライザでLEDを駆動しているI/Oピン(GPIOのP4.7)の状態をモニタしてみましょう．「View」→「Logic Analyzer Window」を選択すると，ロジック・アナライザが起動します(図8の⑥を参照)．ロジック・アナライザのウィンドウが表示されるので［Setup ...］というボタンをクリックし，設定を行います．Setup Logic Analyzerウィンドウの New (Insert) アイコンをクリックし，モニタするレジスタ名「GP4DAT」をタイプした後，Display Typeを「Bit」にします．マスク(And Mask)とビット・シフト(Shift Right)を図8の⑩のように設定し，設定を終えたら［Close］ボタンを押します．

プログラムを実行するとP4.7の状態が描かれます(Logic Analyzerウィンドウのズーム・ボタン［In］，［Out］，［All］で横軸を調整する)．P4.7がタイマ割り込みに同期して，1秒周期でON/OFFすることがわかります(図8の⑪)．

4）実機による動作

最後に，作業フォルダ(C:\ADuC702x\code\DWM_ADuC7026_コード\デジタル電圧計)のdemo.hexをARMWSDで付属基板にダウンロードし，実際に動作させてみます．前述のLED点滅プログラムの場合と同じ手順でプログラムのダウンロードを終えたら，Windowsの「スタート」メニューから「すべてのプログラム」→「アクセサリ」→「通信」→「ハイパーターミナル」を選択し，ターミナル・ソフトウェアを起動します．次にターミナル・ソフトウェアのプロパティで接続を図9のように設定します．

ハイパーターミナルの接続設定の後，付属基板にダウンロードしたプログラムを実行すると，ADC0端子のA-D変換結果が電圧で表示されます．A-Dコンバータは12ビット精度(1LSB　610 μ V)なので，電圧計としての誤差は0～2.5Vレンジで±1mV以下です．

3. IAR Embedded Workbench Kickstart for ARMの使いかた

次に，統合開発環境「IAR Embedded Workbench Kickstart for ARM」の使いかたを解説します．ツールのインストールを行った後，ソフトウェアをC言語で記述し，コンパイルおよびビルドします．コンパイルしたプログラムはARMWSDを使って付属基板にダウンロードし，実動作を確認します．

● インストール

IAR Embedded Workbench Kickstart for ARMのイ

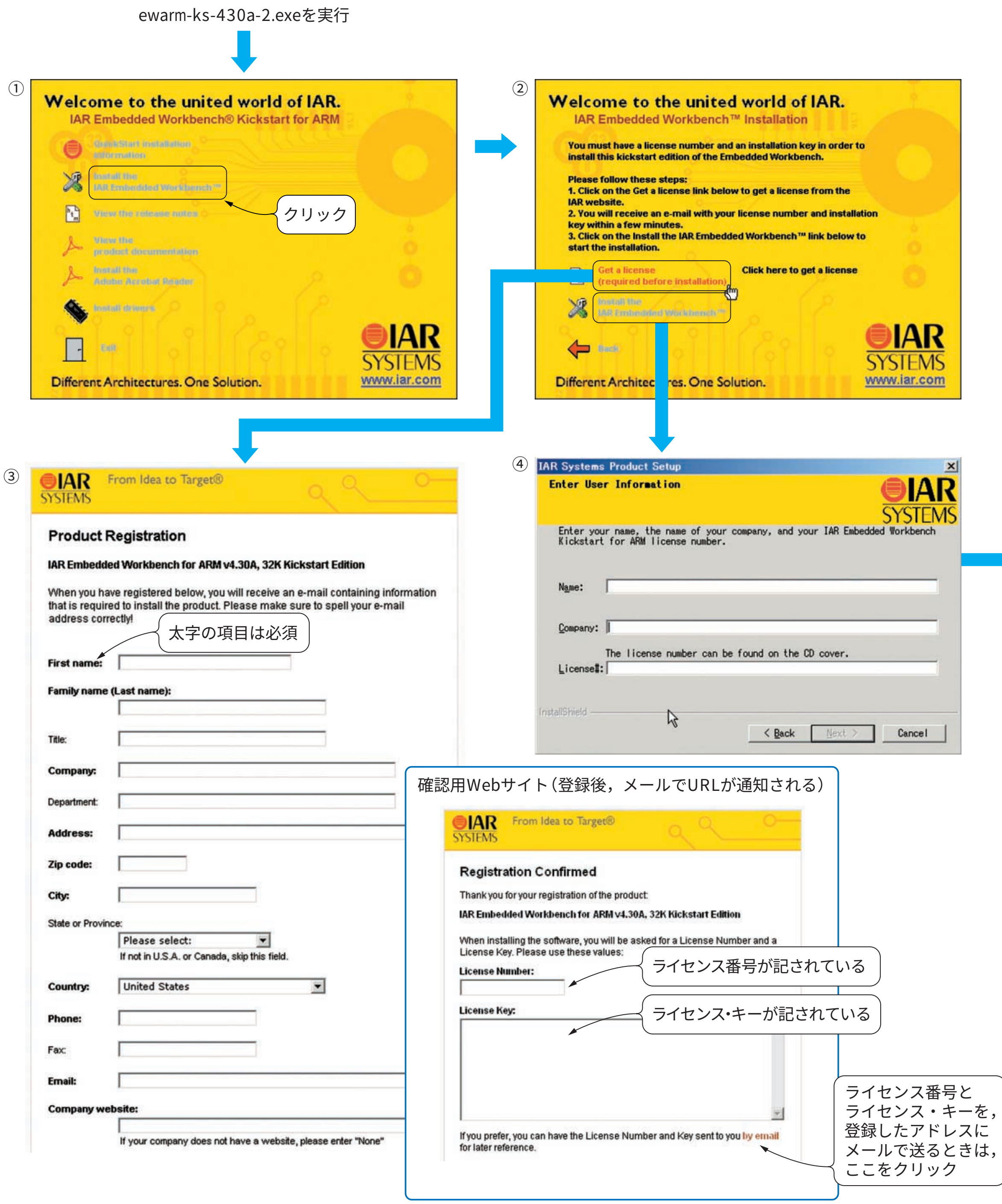


図10　IAR Embedded Workbench Kickstart for ARMのインストール手順

ンストール手順を図10に示します．

まず，付属CD-ROMのsoft\adi\ADuC702xV1.0\IARフォルダにあるewarm-ks-430a-2.exeを実行します．

メニュー画面が表示されたら，「Install the IAR Embedded Workbench」をクリックします（図10の①）．

初めにライセンスを入手します．「Get a license」をクリックするとIARのWebサイトに接続されます（図10の②）．ここでは，パソコンがインターネットに接続されている必要があります．Webページが表示されたら，ユーザ情報と有効なE-mailアドレスを登録します（図10の③）．その後，

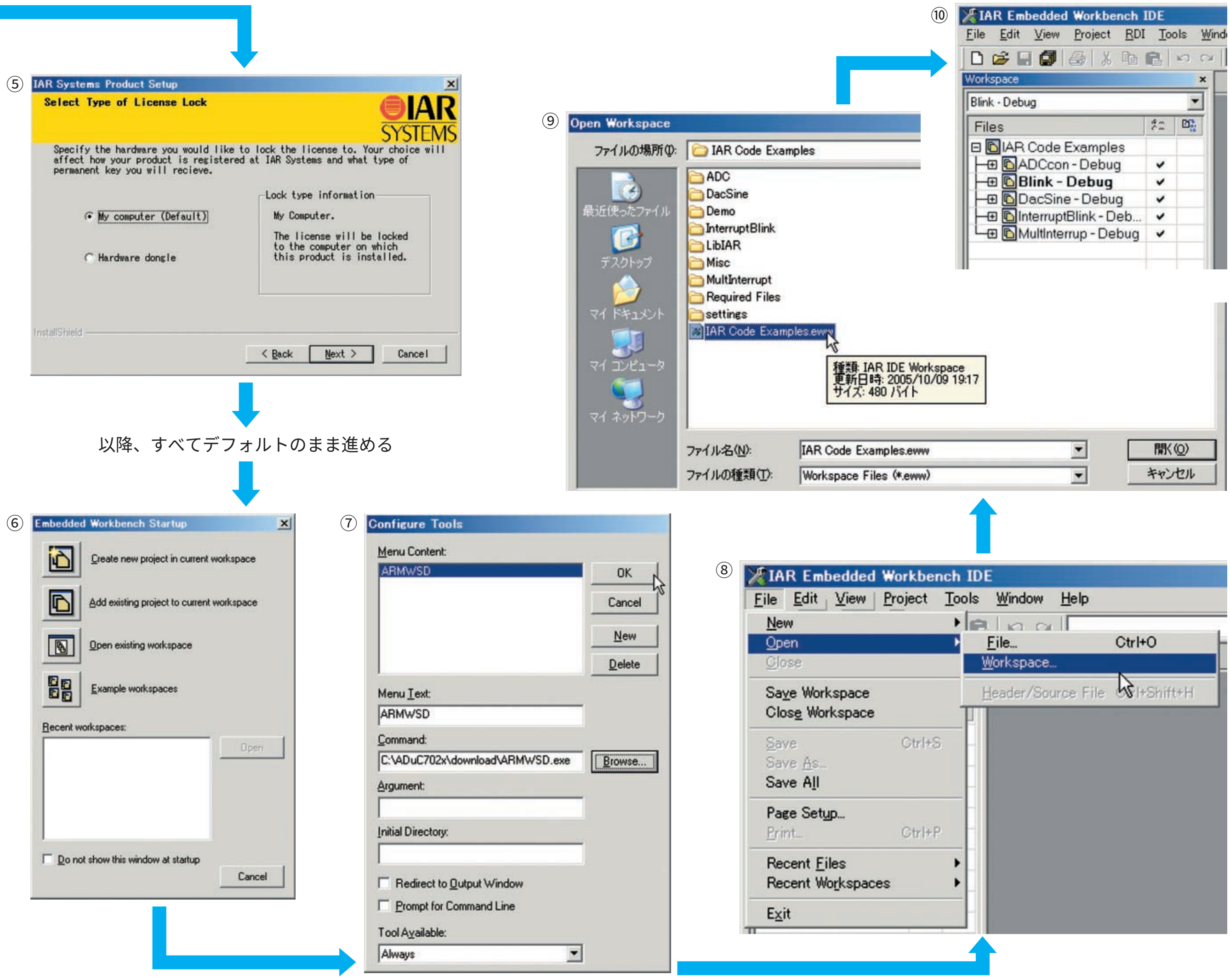


図10　IAR Embedded Workbench Kickstart for ARMのインストール手順(つづき)

登録したE-mailアドレスに確認のメールが送られてきます．メールには，登録確認用WebサイトのURLが記されています(このメールを受け取ってから14日以内にそのURLで確認を行うこと)．このWebサイトで，ライセンス番号とライセンス・キーを取得します．

ライセンスが取得できたら，「Install the IAR Embedded Workbench」をクリックしてインストールを開始します．

ユーザ情報の画面でライセンス番号を入力します(図10の④)．その後ライセンス・キーも入力します．

Select Type of License Lock画面では，デフォルトの「My computer」を選択し，［Next >］ボタンをクリックします(図10の⑤)．この後，インストール先とプログラム・フォルダの登録先を指定しますが，すべてデフォルトのままにして進めてください．

プログラムのインストールが終了すると，IAR Embedded Workbench Kickstart for ARMが自動で起動します．Windowsの「スタート」メニューから「すべてのプログラム」→「IAR Systems」を選択しても起動できます．

起動後，スタートアップ画面が表示されます．あらかじめツールのセットアップを行う必要があるので，ここでは［Cancel］ボタンをクリックします(図10の⑥)．

「Tool」メニューから「Configure Tools ...」を選択します．Configure Toolsウィンドウの［New］ボタンをクリックし，「Menu Text」の欄にダウンローダの名まえである「ARMWSD」を入力します．その後，［Browse］ボタンをクリックし，

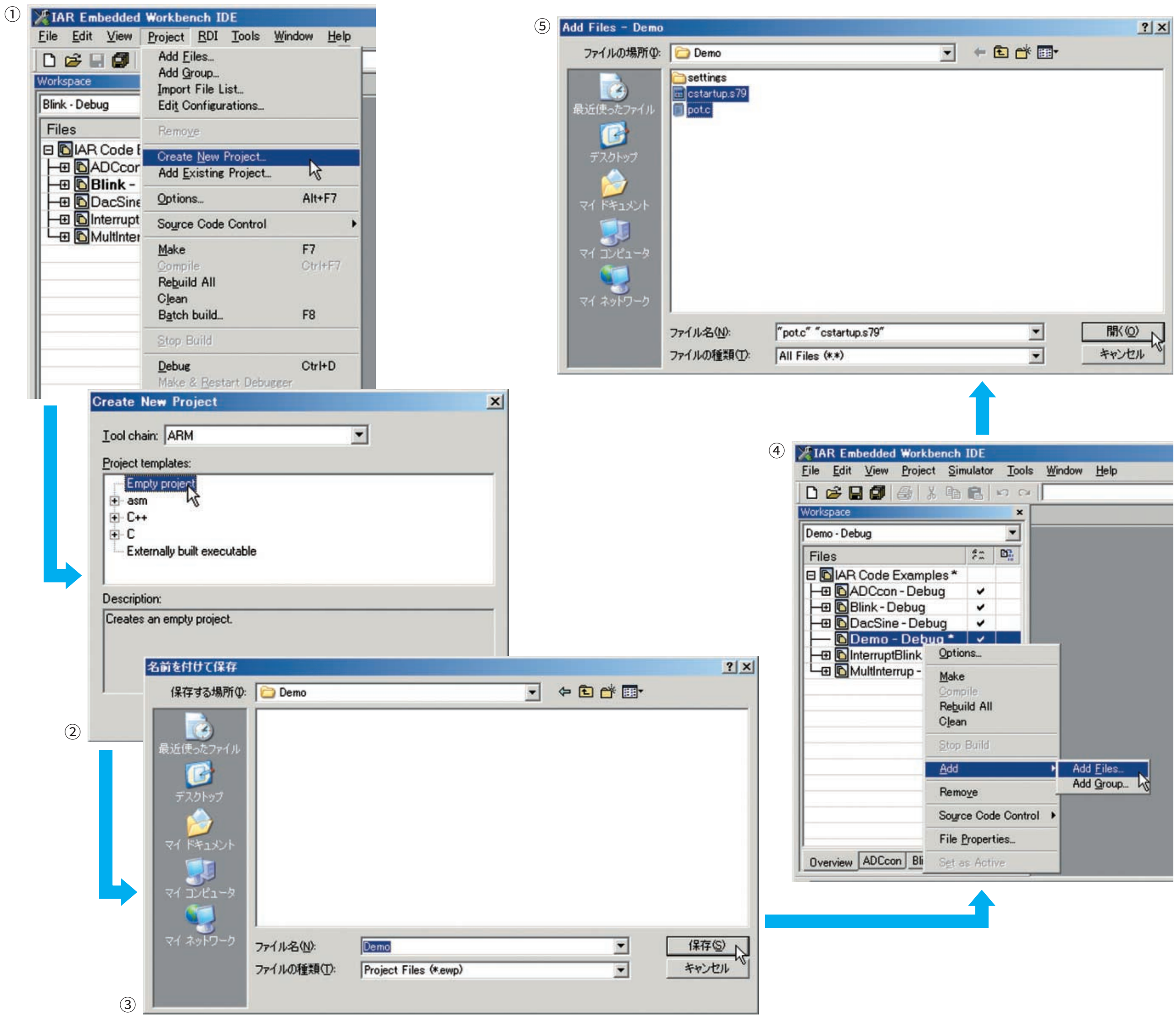


図11　IAR Embedded Workbench Kickstart for ARMのプロジェクトの作成手順

ARMWSDのパスC:\ADuC702x\download\ARMWSD.exeを指定します(**図10**の⑦)．同じようにPLA Toolも登録しておきます．

「File」→「Open」→「Workspace」を選択し，ADuC702x用のサンプル・コード集を開きます(**図10**の⑧)．サンプル・コードはC:\ADuC702x\code\IAR Code Examplesにあります．「IAR Code Examples.eww」を選び，［開く］ボタンをクリックします(**図10**の⑨)．これで，Workspaceウィンドウにサンプル・コードのプロジェクトが一覧で表示されるようになります(**図10**の⑩)．

● プロジェクトの作成

まず，設計データを管理するためのプロジェクトを作成します．手順を**図11**に示します．

新規プロジェクトの作成の場合は，「Project」→「Create New Project」を選択します(**図11**の①)．「Empty project」を選び，［OK］ボタンをクリックします(**図11**の②)．

プロジェクトの保存先を指定します．プロジェクトのファイル名を「Demo」として，C:\ADuC702x\code\IAR Code Examples\Demoに保存します(**図11**の③)．

Workspaceに「Demo」というプロジェクトが追加されます．次に，プロジェクトにソース・ファイルを追加します．

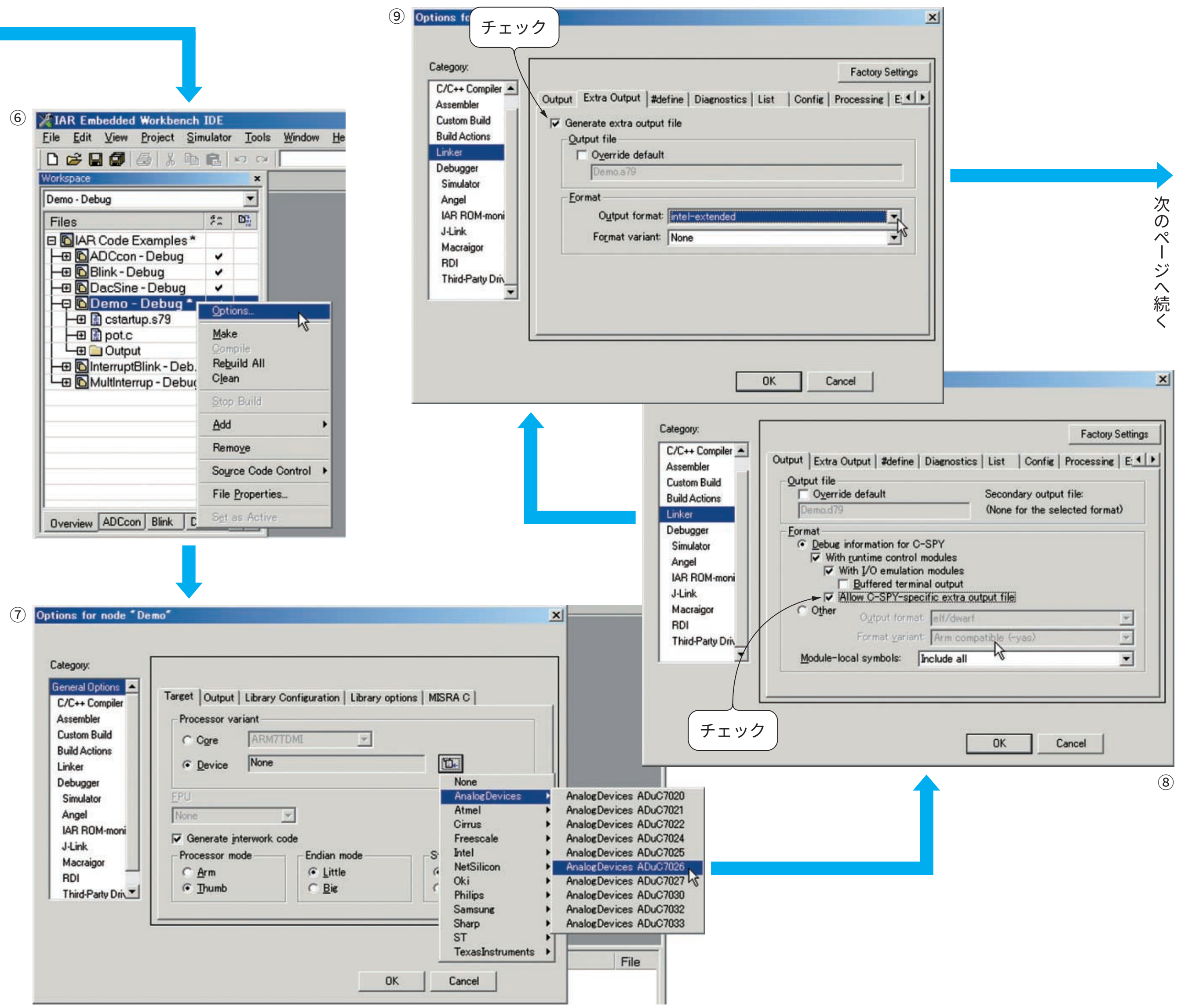


図11　IAR Embedded Workbench Kickstart for ARMのプロジェクトの作成手順（つづき）

Workspaceウィンドウの「Demo」のところを右クリックし，「Add」→「Add Files」を選択します（図11の④）．

Demoフォルダにはあらかじめ，スタートアップ・ファイルとソース・コードが用意されています．cstartup.s79とpot.cを選択して，［開く］ボタンをクリックします（図11の⑤）．ここでファイルの種類を「All Files（*.*）」にしないと，cstartup.s79ファイルは表示されないので，注意してください．

次に，オプションの設定を行います．「Demo」のところを右クリックして「Option」を選択します（図11の⑥）．

オプション画面の左にあるCategory欄からGeneral Optionsを選択します．Targetタブの「Processor variant」を「Device」にして，右にあるボタンから「Analog Devices」→「ADuC7026」を選択します（図11の⑦）．

今度はCategory欄から「Linker」を選択し，Outputタブを表示させ，「Allow C-SPY-specific extra output file」にチェックを入れます（図11の⑧）．Extra Outputタブでは，「Generate extra output file」にチェックを入れます．ファイルのフォーマットを「Intel-extended」にします（図11の⑨）．Configタブでは，「Override default」にチェックを入れます（図11の⑩）．そして［...］ボタンでリンカ・コマンド・ファイルを指定します（図11の⑪）．ファイルはC:\

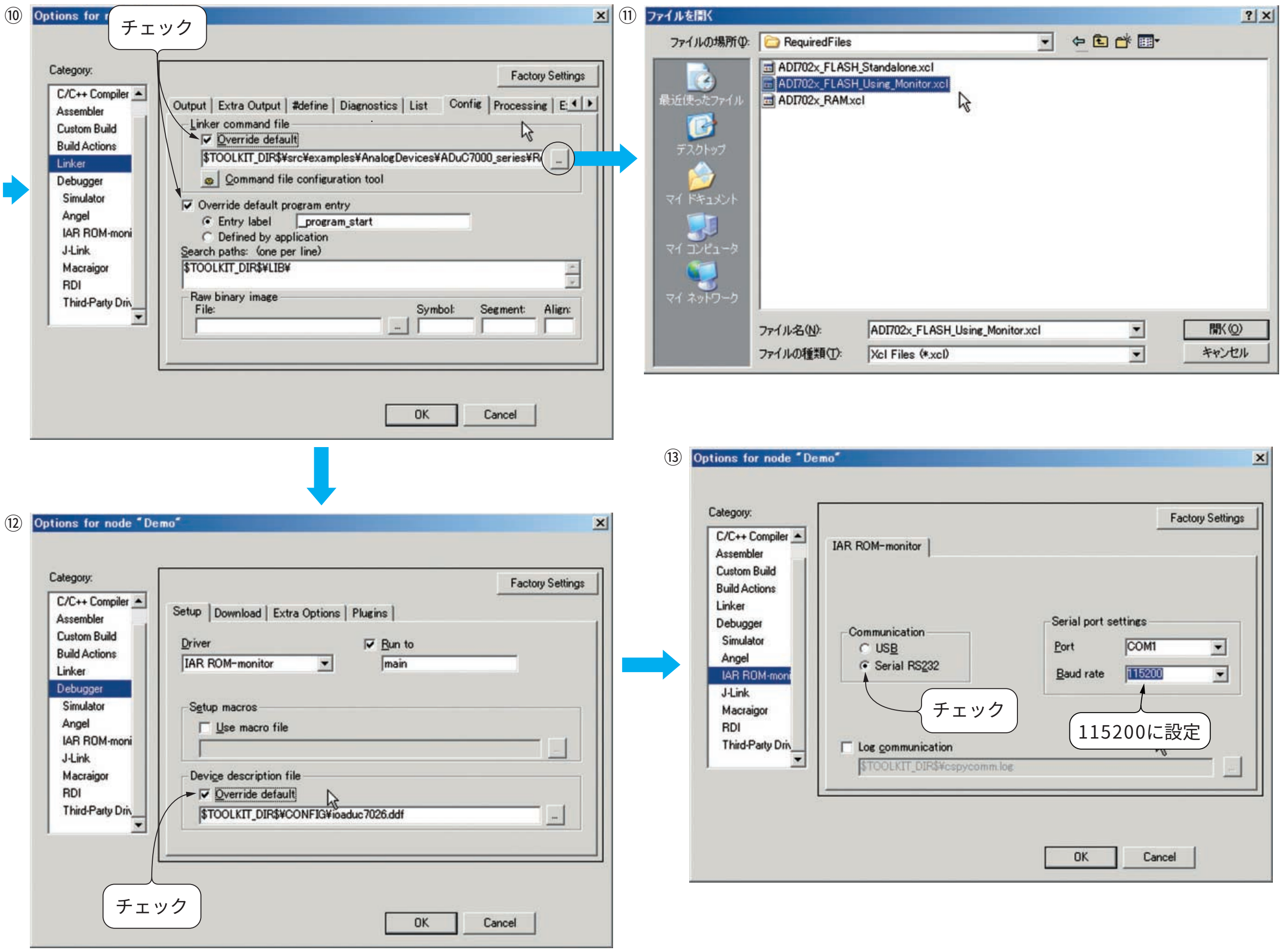


図11　IAR Embedded Workbench Kickstart for ARMのプロジェクトの作成手順（つづき）

Program Files\IAR Systems\Embedded Workbench4.0 Kickstart\arm\src\examples\AnalogDevices\ADuC 7000_series\RequiredFilesのフォルダにあり，ROMモニタを使う場合はADI702x_FLASH_Using_Monitor.xclを指定します（ROMモニタを使わない場合はADI702x_FLASH_Standalone.xclを指定）．さらに「Override default program entry」にもチェックを入れます（**図11**の⑩）．

続いて，デバッガの設定を行います．Category欄から「Debugger」を選択し，Setupタブの「Driver」で「IAR ROM-monitor」を選び，「Override default」にチェックを入れます（**図11**の⑫）．

最後にROMモニタの設定を行います．Category欄から「IAR ROM-monitor」を選択します．「Communication」で「Serial RS232」をチェックし，「Baud Rate」を「115200」に設定します（**図11**の⑬）．［OK］ボタンをクリックして，プロジェクトのオプション設定は完了します．

● ソース・ファイルの登録とコンパイル

作成したプロジェクトにソース・コードを登録し，コンパイルを行います．

Workspaceウィンドウに表示されたpot.cをダブルクリックすると，ソース・ファイルが表示されます（**図12**）．

「Project」→「Compile」でコンパイラを実行します．エラーがなければ，「Project」→「Make」でオブジェクト・コードを作成します．Make後，二つのファイルがプロジェクトのローカル・フォルダのDebug\exeフォルダに生成されます．生成されたファイルは，demo.d79とdemo.a79です．*.a79というファイルは，先ほどリンカのエキストラ・ファイル出力で指定したHEX形式であるため，ファイルの拡張子を.hexに変更しておきます．

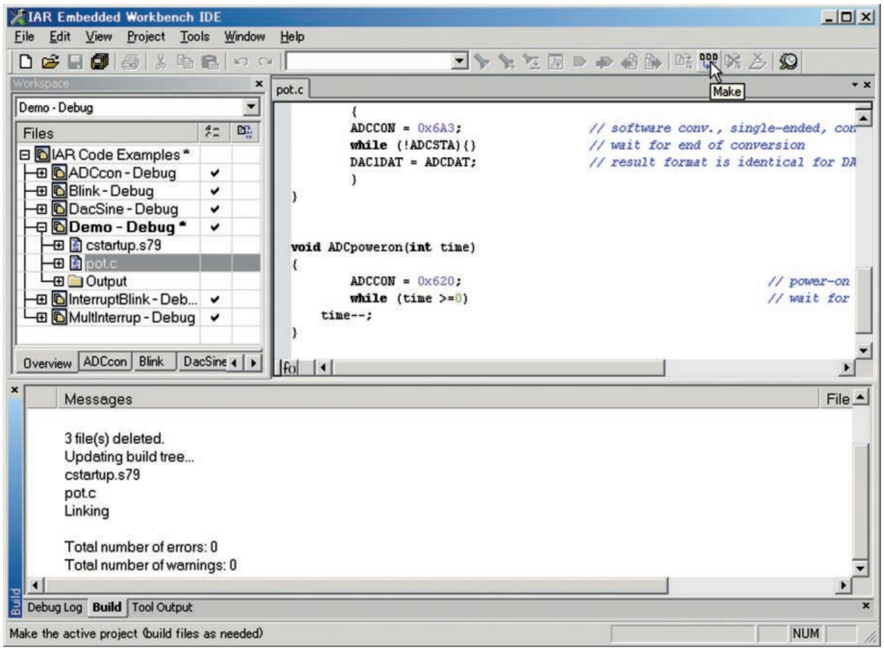


図12
IAR Embedded Workbench Kickstart for ARMのプロジェクトへのコンパイル

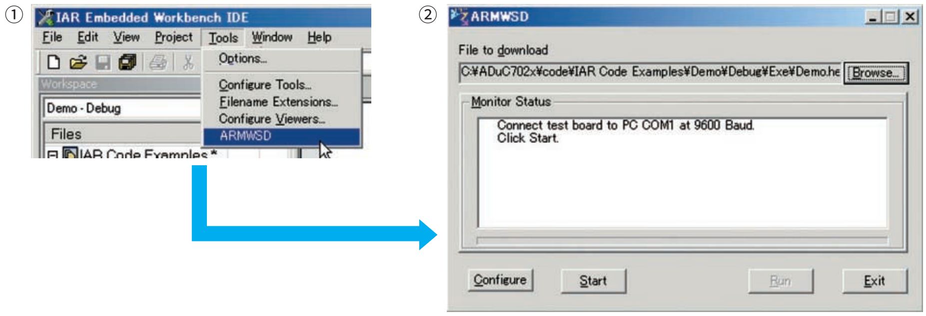


図13
IAR Embedded Workbench Kickstart for ARMからのダウンロード

● ソフトウェアのデバイスへの書き込み

生成されたROMイメージのダウンロードを行います．図7で示した手順のほか，IAR Embedded Workbench Kickstart for ARMでは，メニューからダウンローダを起動できます．

「Tool」→「ARMWSD」を選択すると，ARMWSDが起動します（図13）．ARMWSDの使いかたは，図7と同じです．［Browse］ボタンをクリックし，Makeで作成したDemo.hex（Demo.a79の拡張子を変更したファイル）を選びます．

ADuC7026をダウンロード・モードにセットし，［Start］ボタンをクリックすると，ダウンロードが開始されます．ダウンロード完了後，ADuC7026をリセットする（SW1を押す）とプログラムを実行します．今回のデモでは，ADC0の電圧がDAC0に出力されます．

● ソース・レベル・デバッグ

IAR Embedded Workbench Kickstart for ARMでは，JTAG経由，シミュレータ，ROMモニタのいずれかによってソース・レベル・デバッグが行えます．JTAG経由のデバッグにはJTAGエミュレータ（mIDAS-Linkなど）が必要です．

ここではROMモニタを使った実機によるデバッグについて説明します．図14に手順を示します．

まず，「Tool」→「ARMWSD」を選択してARMWSDを起動します．ADuC7026をダウンロード・モードにして，ROMモニタADIRomMonitor.hexをダウンロードします（図14の①）．ROMモニタはADuC702x\code\IAR code Examples\Required Filesディレクトリにあります．

ROMモニタをダウンロードした後，かならずADuC7026をリセットし，モニタ・プログラムを実行しておきます．

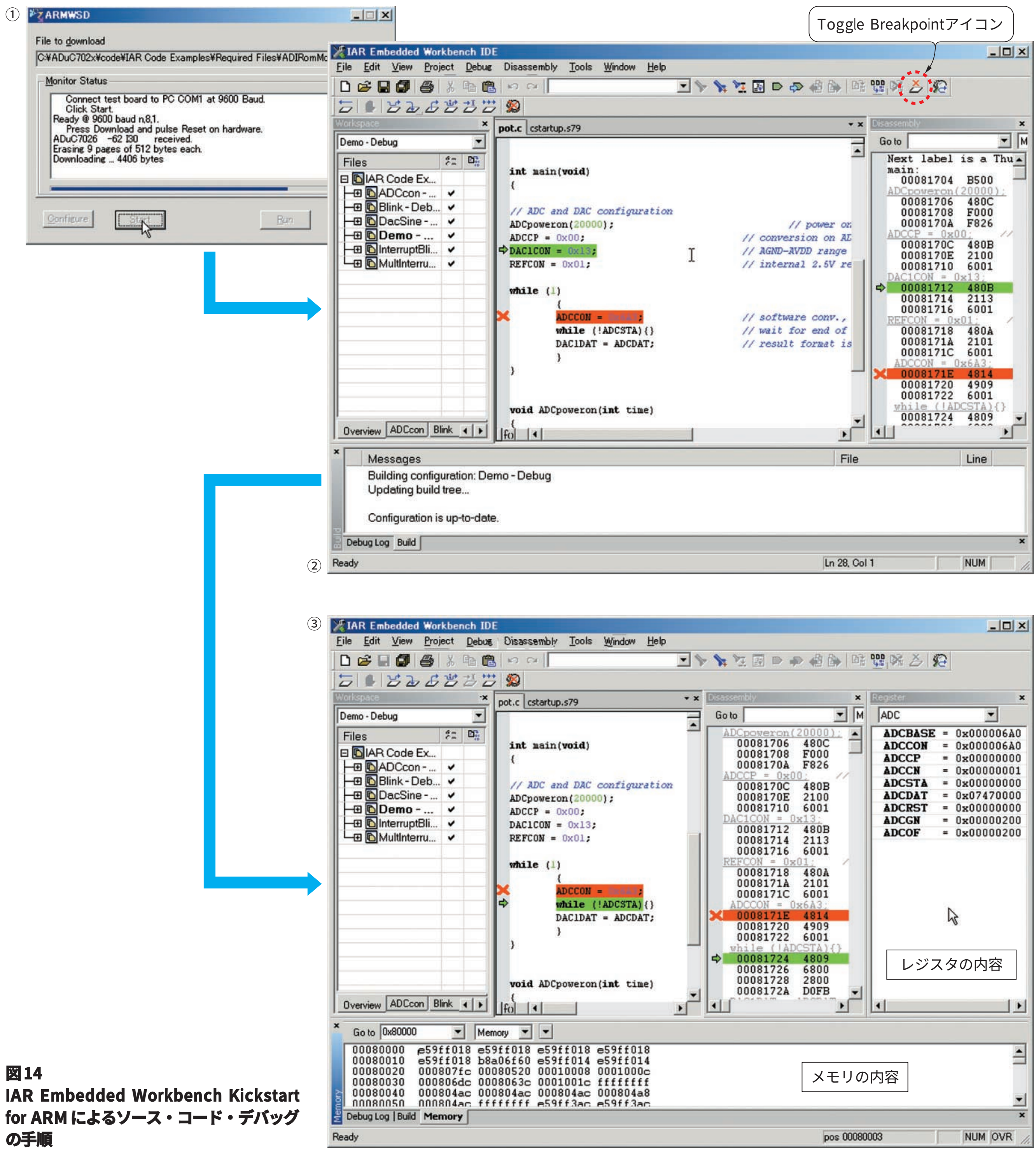


図14
IAR Embedded Workbench Kickstart for ARMによるソース・コード・デバッグの手順

次に「Project」→「Debug」でデバッグ・セッションを開始します．ブレークポイントはToggle Breakpointアイコンをクリックして設定します(**図14**の②)．

ADuC7026が内蔵する周辺I/Oは「View」→「Register」で，フラッシュ・メモリやRAMは「View」→「Memory」で表示されます(**図14**の③)．RegisterウィンドウでADCを選択すると，A-Dコンバータのレジスタを表示できます．

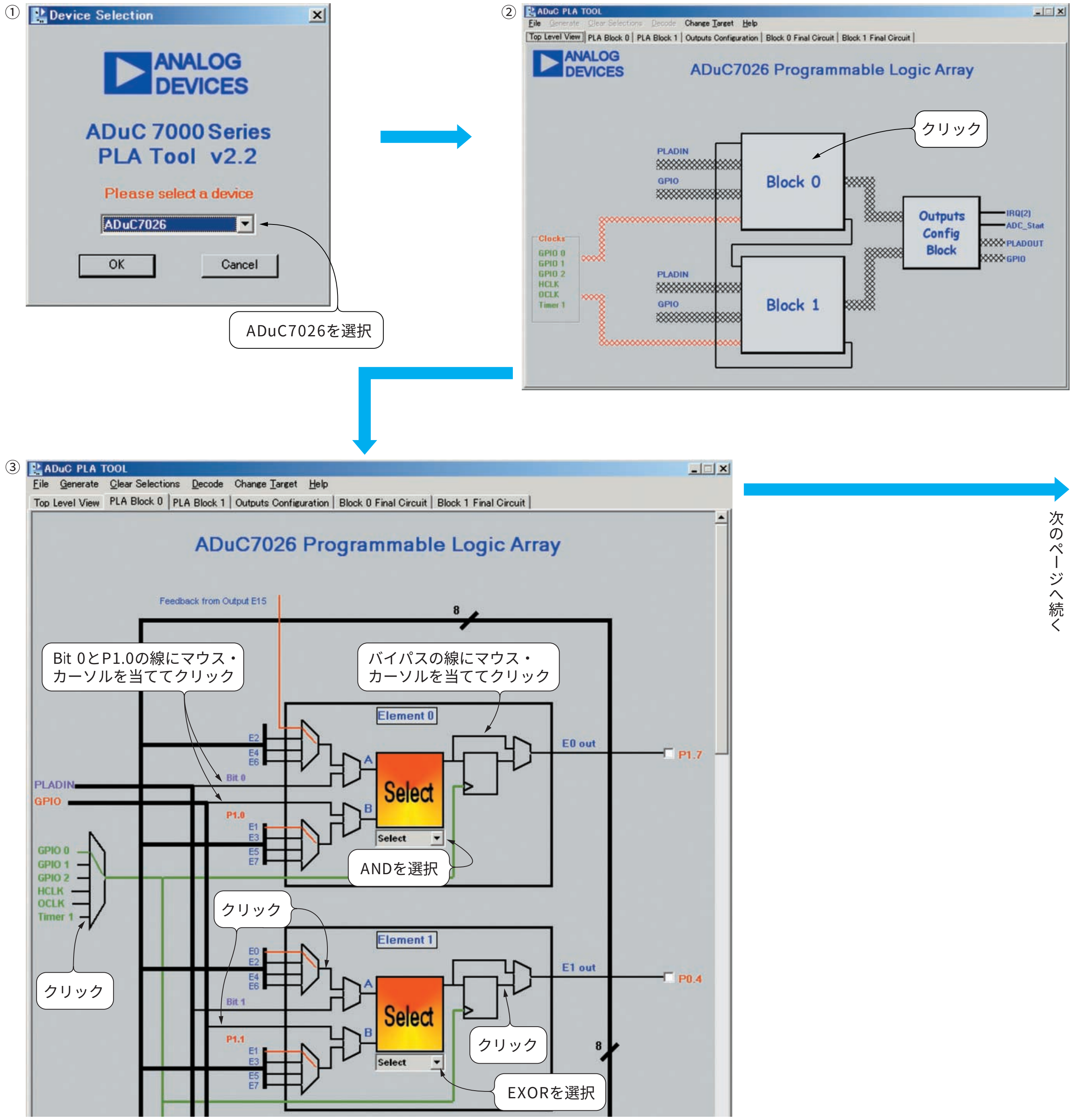


図15　PLA Toolの使いかた

次のページへ続く

4

4．PLA設計ツールの使いかた

ADuC7000シリーズは8エレメント（Element），2ブロック（Block）構成のプログラマブル・ロジック・アレイ（PLA）を内蔵しており，合計16エレメントを使うことができます．PLAの設計には専用のPLA Toolを使用します．ここでは，PLA Toolの使いかたについて説明します．

ブロック0の2個のエレメントをPLA Toolを使って設定し，C言語とアセンブリ言語のコードを生成します．手順を図15に示します．

Windowsの「スタート」メニューから「すべてのプログラ

④

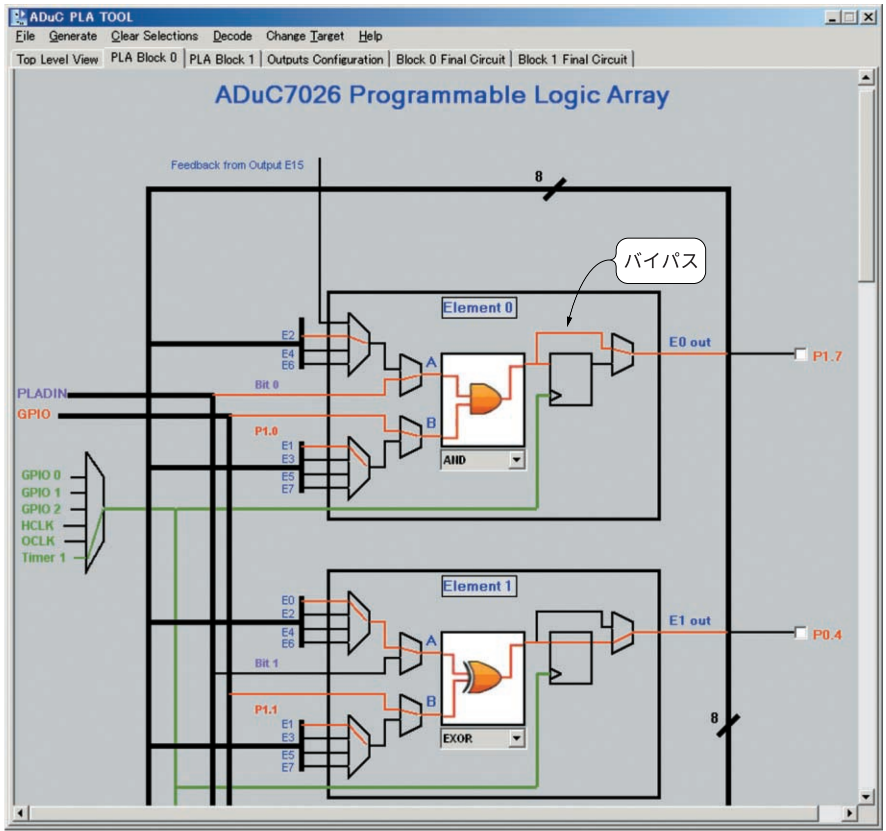


図15　PLA Toolの使いかた(つづき)

ム」→「ADuC702x」→「PLA Tool」を選択し，PLA Toolを起動します．

デバイス選択画面が表示されます．ADuC7026を選択し，[OK]をクリックします(**図15**の①)．

PLAの全体図が表示されます．今回はBlock 0を使うので「Block 0」をクリックします(**図15**の②)．

マウスを使って，各エレメントのマルチプレクサに入力する信号を選択していきます(**図15**の③)．

Element 0に対しては，PLADINレジスタのBit 0の信号とGPIOのP1.0を選択します．線の上をクリックすると赤くなり，選択されたことになります．線が細いため，少しずれていると赤く変わらないので注意してください．ルックアップ・テーブル(Element 0の中央の四角)のプルダウン・メニューから「AND」を選択します．Element 0のフリップフロップはバイパス(フリップフロップを通らない配線)です(**図15**の④)．

Element 1に対して，Element 0(E0)の出力とGPIOのP1.1を選択し，ルックアップ・テーブルは「EXOR」を選択します．Element 1はフリップフロップを利用します．フリップフロップのクロックはTimer 1を選択します．Timer 1のすぐ横の配線をクリックすると，緑色に変わります．

Outputs Configurationのタブをクリックします．ここで各エレメントの出力をGPIOに出力することができます．ただし，CPUからはPLADOUTレジスタ経由(AMBAバス)で全16エレメントの出力をモニタできるため，ここでのGPIO設定はかならずしも必要ありません．外部デバイスを制御する場合に利用します．「GPIO Output Pins」で対応するボックスにチェックを入れると，GPIOへの出力が有効になります．ここではElement 1の出力をP0.4に反映させるようにしました(**図15**の⑤)．PLAの割り込み要因として，PLAIRQ0とPLAIRQ1を利用できます．ここではElement 1の出力をPLAIRQ0へ接続しました(**図15**の⑥)．エレメントの出力をA-D変換の開始トリガに使うこともできます(**図15**の⑦)．A-D変換開始信号は負論理なので，必要に応じてEXORをEXNORにします．

Block 0 Final Circuitタブをクリックすると，Block 0の最終回路図が表示されます(**図15**の⑧)．問題がなければこれで設定は終了です．

次のページへ続く

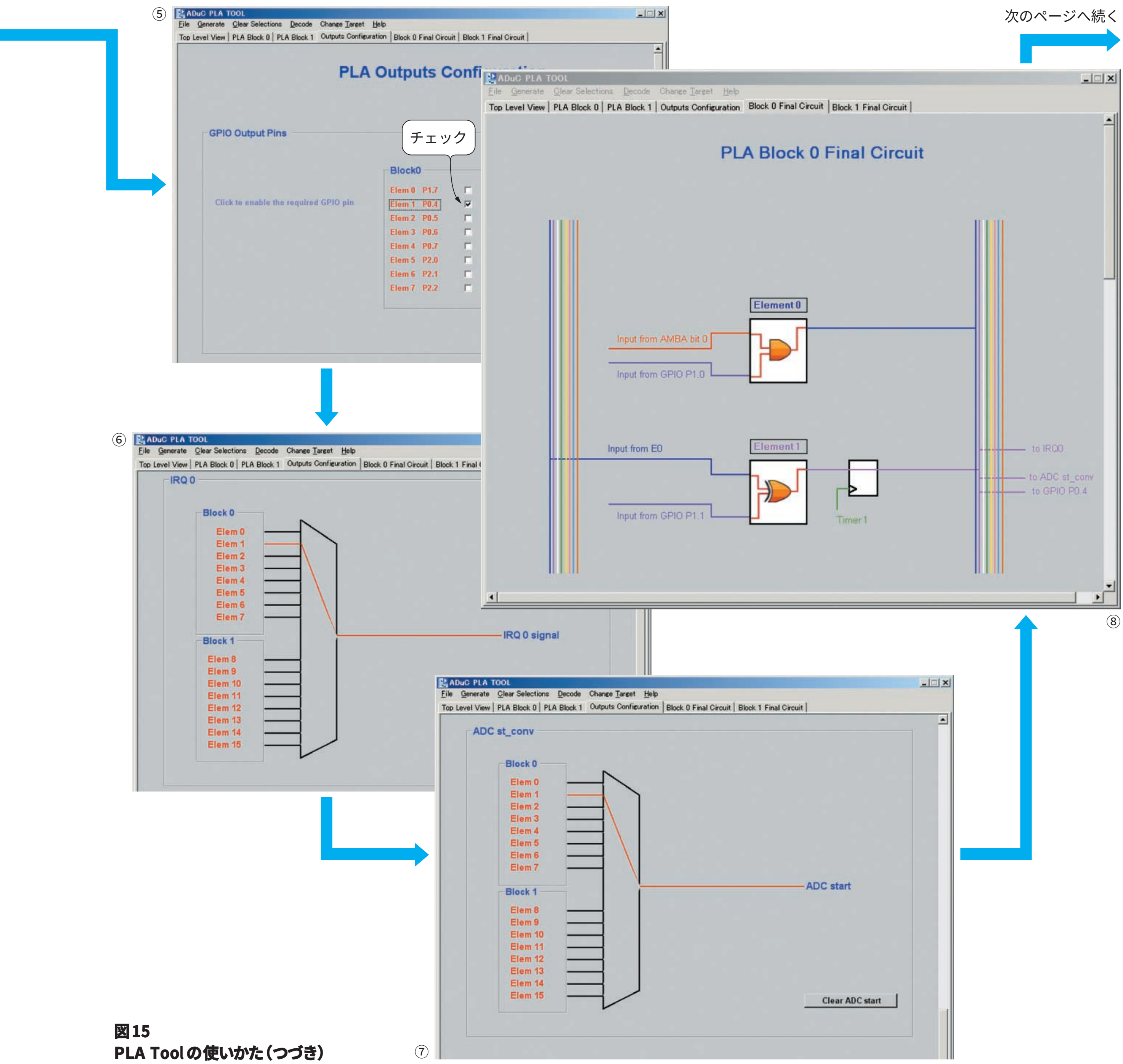


図15
PLA Toolの使いかた(つづき)

「Generate」→「Register Values」を選択すると，PLA Memory Mapped Register Values画面が表示されます．各エレメントのレジスタ設定値が16進数で表示されます(**図15**の⑨)．ユーザ・プログラムでElement 0に0x31，Element 1に0X6Cを書き込めばよいことがわかります．

Clock Source，ADC and IRQ registersタブでは，クロック・ソースとA-D変換トリガ・ソース，IRQソースを確認できます(**図15**の⑩)．

「Generate Code」を選択すると，アセンブリ・コードとCソース・コードが生成されます(**図15**の⑪)．これをユーザ・プログラムにカット・アンド・ペーストして使います．

まつぞえ・のぶひろ
アナログ・デバイセズ(株)

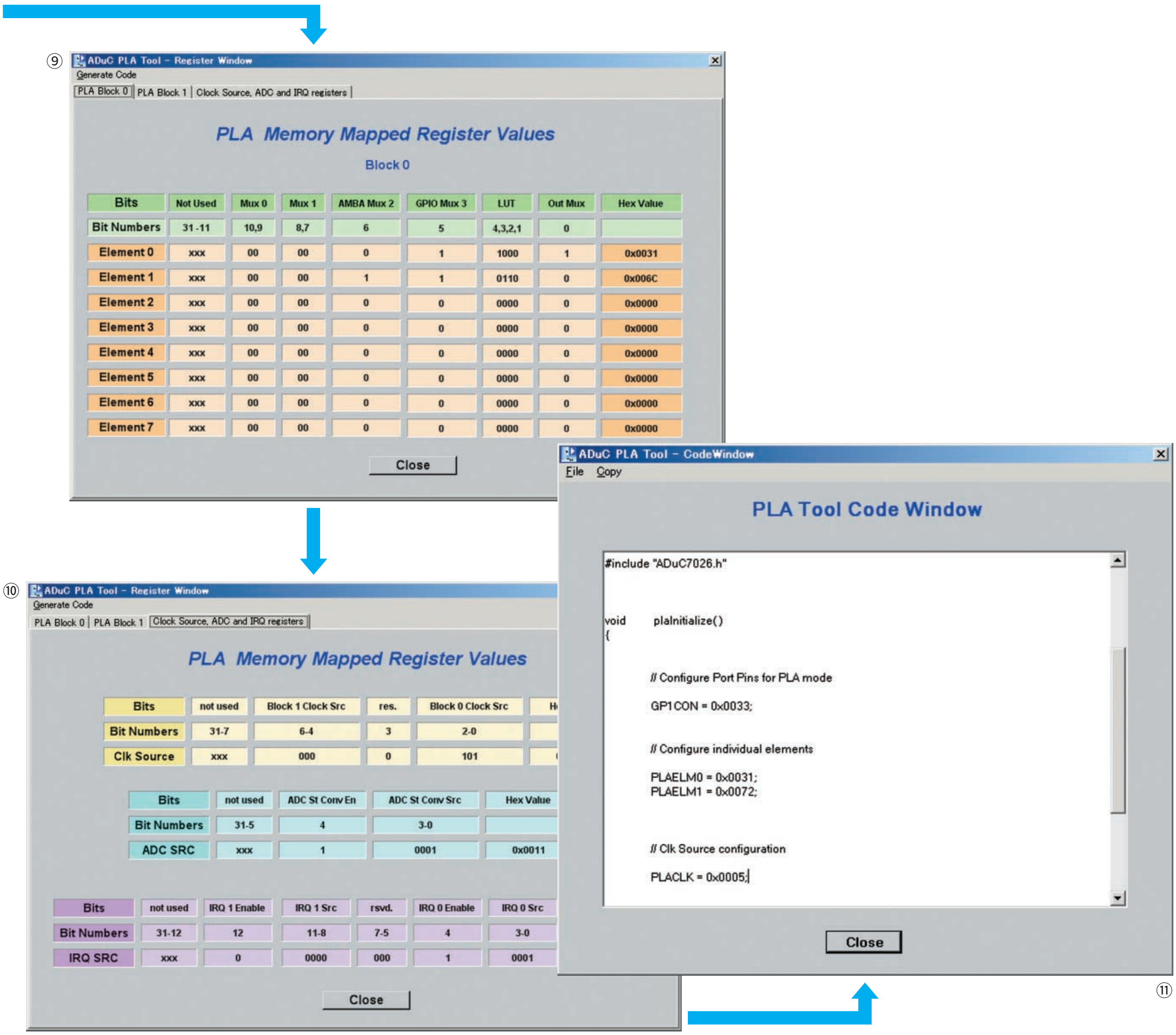


図15　PLA Toolの使いかた（つづき）